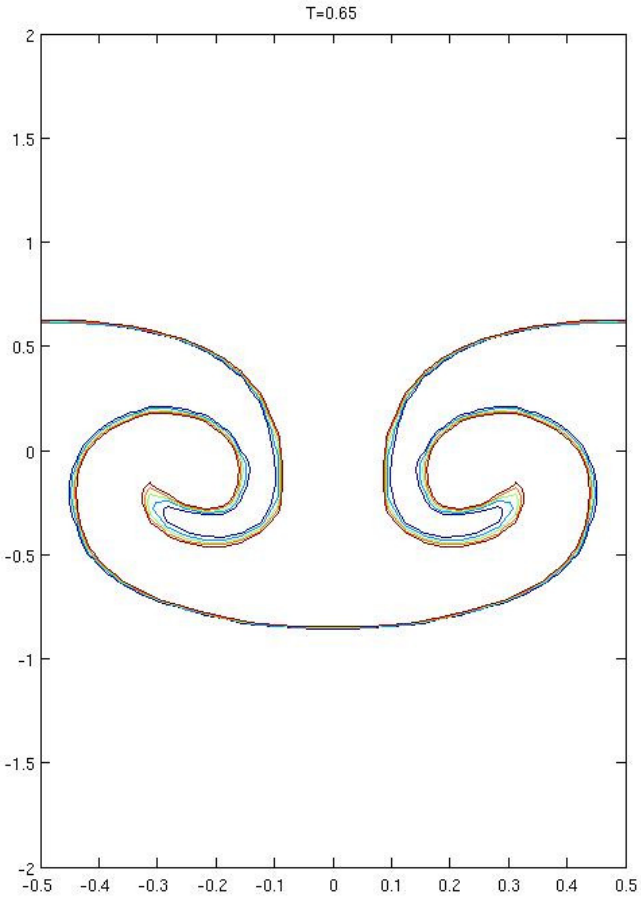
NS2DVD
Navier Stokes 2D Variable Density



User guide

Laboratoire
Paul Painlevé

Contents

I Introduction **2**

1 General explanations **2**

2 Description of the numerical scheme **2**
2.1 Solving the density equation by a Finite Volume method 2
2.2 Solving the velocity equation by a Finite Element method 2

II Settings **2**

3 Setup introduction **3**
3.1 Two Possiblities 3
3.2 Directories and files 3

4 Graphic User Interface **3**
4.1 Mesh settings 4
4.1.1 Domain geometry 4
4.1.2 Mesh design 4
4.2 Physical parameters 5
4.3 Finite Volumes parameters 5
4.3.1 Control cells design 6
4.3.2 FV scheme 7
4.4 Finite Elements parameters 7
4.4.1 Constrained optimisation 8
4.4.2 Projection method 8
4.5 Computation parameters 8
4.6 Outputs parameters 9

5 Advanced settings **9**
5.1 Pressure constraint 9
5.2 Advanced FV parameters 10
5.2.1 Flux limiter 10
5.2.2 FV step time 10
5.3 Advanced FE parameters 10
5.4 Rerun computation 11

6 Post-processing **11**
6.1 Convergence rates 11
6.2 Animation 11

7 Benchmark cases **11**
7.1 Validation benchmarks 11
7.1.1 Analytical Benchmark (EXAC) 11
7.1.2 Lid Driven Cavity (DCAV) 12
7.1.3 Density Gaussian Translation (TRHO) 13
7.1.4 Density Gaussian Rotation (RRHO) 13
7.2 Other benchmarks 14
7.2.1 Rayleigh-Taylor Instability (RTIN) 14
7.2.2 Falling Droplet (DROP) 15
7.3 New benchmark cases 16

III Appendix **16**

A Splitting time types **16**
A.1 Euler splitting 16
A.2 $(n+1)$ velovity linear second order extrapolation 17
A.3 $(n+1/2)$ velovity linear second order extrapolation 17
A.4 Strang splitting 17

Part I
Introduction

1 General explanations

This code, collectively developped the Paul Painlev Mathematics Laboratory (UMR Lille-1 University / CNRS N 8524) and INRIA Lille Nord Europe, is devoted to the numerical simulation of the variable density incompressible Navier-Stokes system given on a domain $\Omega \subset \mathbb{R}^2$ by :

$$\begin{cases} \partial_t \rho + \mathrm{div}_{\mathbf{x}}(\rho \mathbf{u}) = 0, & \text{(1a)} \\ \rho\big(\partial_t \mathbf{u} + (\mathbf{u}\cdot\nabla_{\mathbf{x}})\mathbf{u}\big) + \nabla_{\mathbf{x}} p - \mu \Delta_{\mathbf{x}} \mathbf{u} = \mathbf{f}, & \text{(1b)} \\ \mathrm{div}_{\mathbf{x}} \mathbf{u} = 0. & \text{(1c)} \end{cases}$$

Here $\rho(t,\mathbf{x}) \geq 0$ stands for the density of a viscous fluid whose velocity field is $\mathbf{u}(t,\mathbf{x}) \in \mathbb{R}^2$ and pressure $p(t,\mathbf{x}) \in \mathbb{R}$. The description of the external force is embodied into the right hand side $\mathbf{f}(t,\mathbf{x})$ of (1b) and $\mu > 0$ is the viscosity of the fluid. The unknowns depend on time $t \geq 0$ and position $\mathbf{x} \in \Omega \subset \mathbb{R}^2$.

2 Description of the numerical scheme

The mass conservation relation (1a) is solved thanks to a Finite Volume (FV) method and the momentum equation (1b) is solved with a Finite Element (FE) method, taking into acount the divergence free constraint (1c). Four different time splittings can be used for solving the system (1a)-(1b)-(1c).

FV part and FE part are solved separately successively, or those part are computed alternately FV - FE, then FE - FV. Those different time splittings give different accuracy orders. Time semi discretisation is detailed in appendix A.

2.1 Solving the density equation by a Finite Volume method

As presented in [3] and [2], the transport equation (1a) is solved with a vertex-based Finite Volume method combined with a MUSCL scheme and the use of of a flux limiter. FV parameters are detailed in section 4.3.

2.2 Solving the velocity equation by a Finite Element method

In the numerical simulation of the Navier-Stokes eq. (1b) a major difficulty is that the velocity and the pressure are coupled by the incompressibility constrain (1c).
In order to solve this system, two types of schemes are implemented in the code :

- Constrained optimization methods.
- Projection methods.

FE parameters are detailed in part 4.4.

Part II
Settings

3 Setup introduction

3.1 Two Possiblities

Several parameters can be set, according to the simulation we want. All of them are regrouped in the file called "manual_setup.m" which is preset for simulating lid driven cavity benchmark (see subsection 7.1.2). It can so directly be run to get a first example of a simulation.
In order to simplify the setup and to exclude incompatible parameters, a Graphic User Interface (GUI) has been developped, more details about this interface are given in section 4.

3.2 Directories and files

Once the archive extracted, move to NS2DVD directory. Each ".m" file contains an eponymous function, most of them are ordered in the folowing sub-directories :

- ASSEMBLING : This directory regroups all fonctions used for assembling matrix for the FE part.
- AUXILIARY_FUNCTIONS : Here are the functions which do not compute anything. For example, the ones used for displaying headers, or for building the Graphic User Interface.
- COMPUTATION : Here are the computation functions, grouped depending on their use (Finite Elements (FE), Finite Volume (FV), etc).
- INITIALISATION : This directory regroups functions used for setting boundary conditions and defining initial interface(s) equation(s) for bi-fluid simulations.
- MESH_BUILDING : Function used for building the mesh or loading an external mesh are grouped in this directory.
- OUTPUTS : Files and data generated during the computation will be stored in this directory.
- POST_PROCESSING : This directory contains functions used for plotting data or making a film after the computation.

The NS2DVD directory also contains the following files :

- GUI_launcher.m : Run this file for starting Graphical User Interface (see section 4).
- manual_setting.m : This file regroups all paramaters and permits to run a computation without using the GUI.
- main_P1P2P1.m : This is the main function, it can be called from the GUI or from the "manual_setup.m" file.
- user_guide.pdf : The present documentation.

4 Graphic User Interface

To run GUI, change directory to NS2DVD and run the "GUI_launcher" function. The first window, which permits to set mesh parameters, corresponds to the part "A" of the "manual_setup.m" file. The second window, which permits to set computation and outputs parameters, corresponds to the part "B".

4.1 Mesh settings

Two kinds of mesh can be built, or a structured one built by hand, either a non structured one, built thanks to the Matlab Partial Differential Equations Toolbox (PDE Toolbox). An unstructured mesh can also be used without using the PDE Toolbox, then mesh data files needs to be previously generated with a mesh editor. Some mesh data files have already been generated for for unit disk and for a $[-0.5, 0.5]\times[-2, 2]$ rectangle domain. Those mesh files are placed in "MESH_BUILDING/MESH_FILES". Create a file which contains P1 nodes coordinates and a second one which contains nodes numbers of each triangle. Mesh design possibilities are summarized in Figure 1 and detailed below :

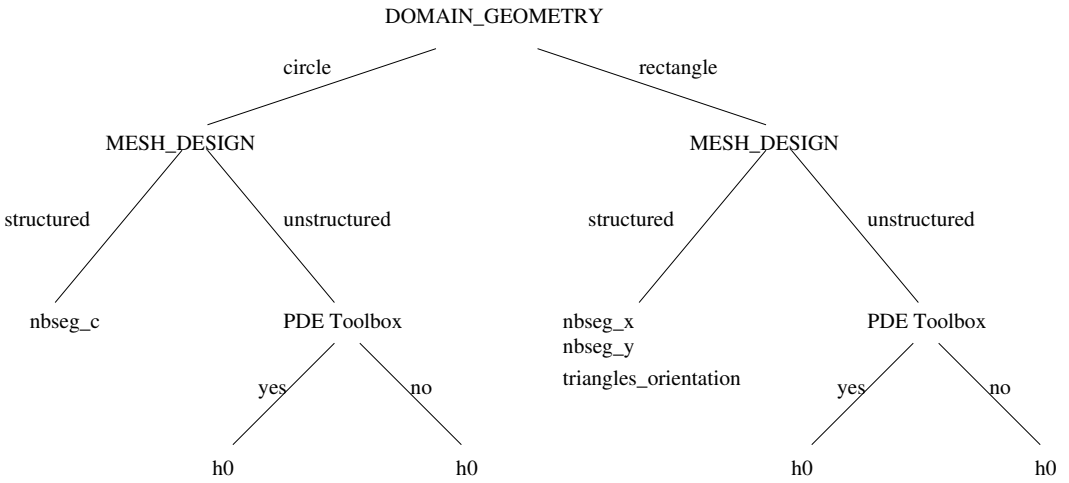


Figure 1: Mesh parameters tree.

4.1.1 Domain geometry

- For a rectangle geometry, set coordinates of the points, taking into account that (x_1, y_1) is the one on the bottom left and (x_2, y_2) the one on the top right.
- For a circle geometry, set the center coordinates and the radius value.

NB : Those parameters are grouped in the "domain" structure.

4.1.2 Mesh design

- For an unstructured mesh, set the maximum edge length h_0.
- For a structured mesh (hand-built), set the number of subsegments on the radius for a circle, or on x and y axis for a rectangle geometry.

  For a rectangle geometry, choose between 2 following triangles orientations :

  > diagonal : each quadrant is subdivised with parallels diagonals as presented in figure 4.1.2-b.

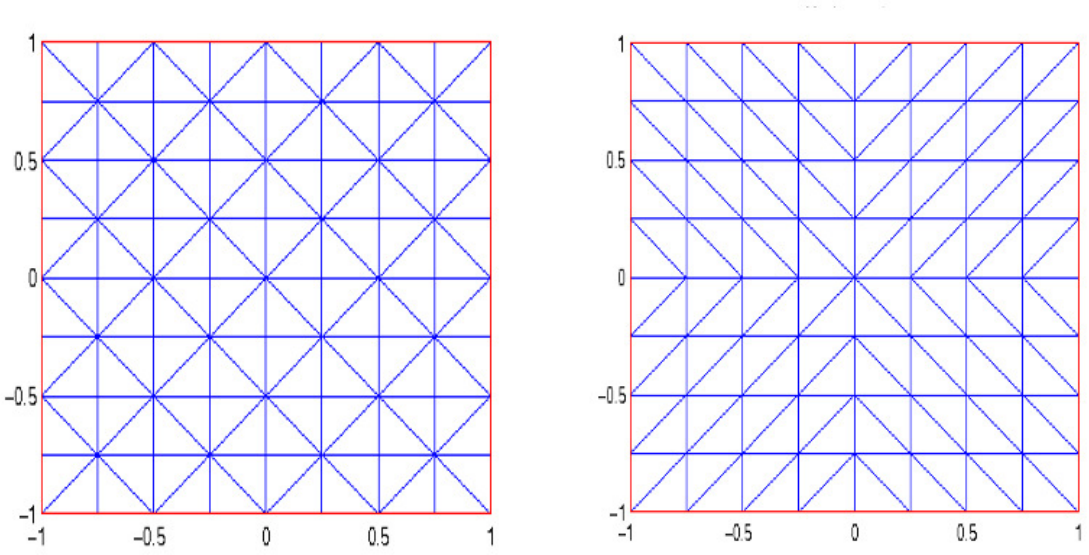
  > cross : each 2×2 rectangle cells are split by and "X" as presented in figure 4.1.2-a.

NB : Those parameters are grouped in the "mesh" structure.

Remark 1 *Depending on the parameters set in the mesh setting window, the edge size parameter "h_{max}" is computed by different ways.*

- *If the domain geometry is a rectangle and the mesh is structured :*

$$h_{max} = max\Big(\frac{x_2 - x_1}{nbseg_x}, \frac{y_2 - y_1}{nbseg_y}\Big)$$



(a) (b)

Figure 2: Structured meshes: (a) -"cross"; (b) -"diagonal"

- *If the domain geometry is a circle and the mesh is structured :*

$$h_{max} = \frac{r_0}{nbseg_c}$$

- *For an unstructured mesh, maximal edge size has already been declared, so :*

$$h_{max} = h_0$$

4.2 Physical parameters

- Reynolds number.
- Final simulation time.
- Gravity value.
- Density
- Maximum density (in case of bi-fluid flow)
- Boundary Conditions (BC)
  - > Velocity first composant BC
  - > Velocity second composant BC
  - > Density BC

Remark 2 *For most of benchmark cases, boundary conditions can't be modified because possibilities are not implemented or in some cases it does not make sense.*

NB : Those parameters are grouped in the "PHYSICAL" structure.

4.3 Finite Volumes parameters

The Finite Volume method implemented in NS2DVD is presented in [3] and [2].

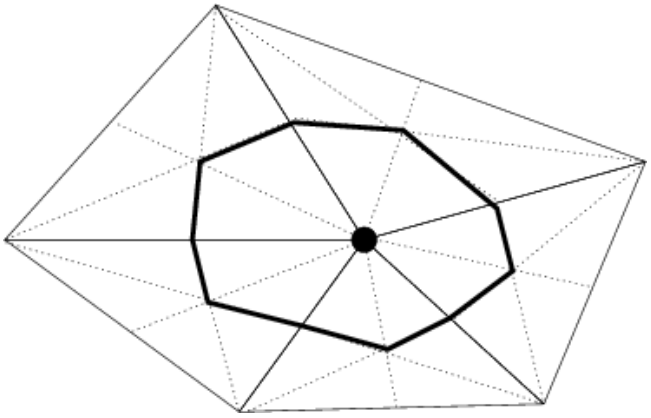


Figure 3: Control cell for unstructured mesh

4.3.1 Control cells design

Control cells of finite volume scheme are vertex centered.

- for an unstructured mesh, control cells are built by linking each barycenter to the center of its neighboring edges as presented in figure 3.

- in case of a structured mesh, for both designs presented in section 4.1.2, two kinds of control cells can be built.

  > Either cells are built as seen previously for an unstructured mesh, then, they are called "star" control volumes. For a "cross" structured mesh (see section 4.1.2), cells are octogonals, and for a "diagonal" structured mesh, cells are hexagonals as presented in figure 4(a).

  > Or cells are built in order to have square control volumes as presented in figure 4(b).

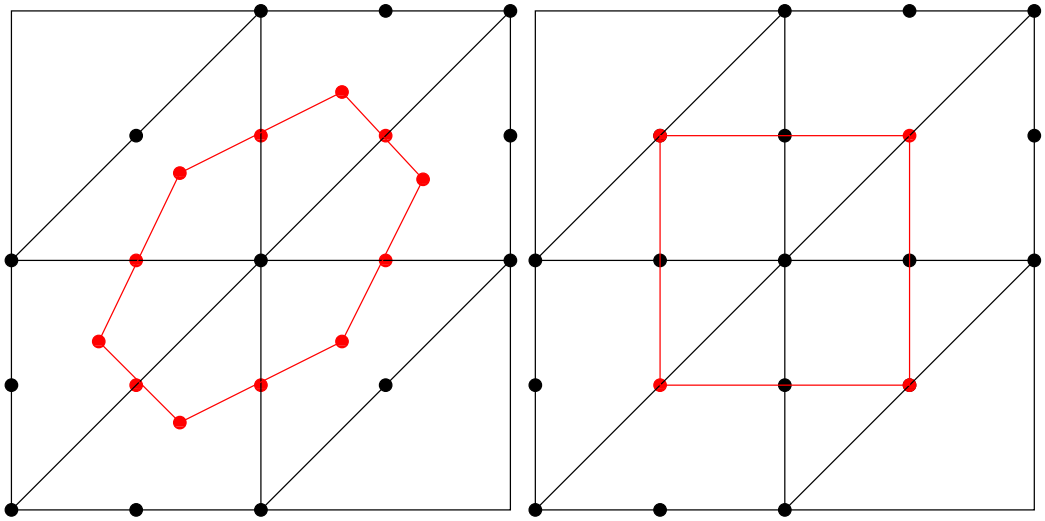


Figure 4: Control volume on structured diagonal mesh : a - Star; b - Square

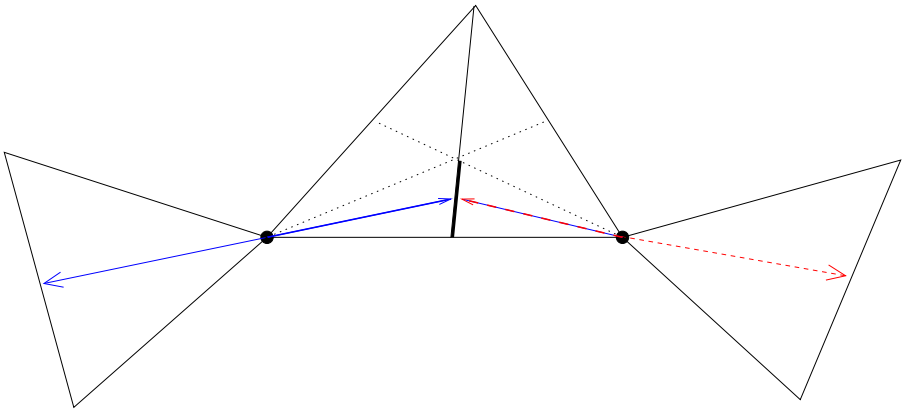


Figure 5: Node identification for computing gradient

4.3.2 FV scheme

- **Beta** is the parameter of the so-called β-scheme. More precisely, for the MUSCL strategy, the gradients $\nabla^- \rho_A$ and $\nabla^+ \rho_A$ are defined as :

$$\nabla^- \rho_A = \beta\, \overline{\nabla \rho_A} + (1-\beta)\, \underline{\nabla^- \rho_A}$$

$$\nabla^+ \rho_A = \beta\, \overline{\nabla \rho_A} + (1-\beta)\, \underline{\nabla^+ \rho_A}$$

with

$$\overline{\nabla \rho_A} = \frac{\sum_{j=1}^{nt} |M_j| (\nabla \rho)_{|_{M_j}}}{\sum_{j=1}^{nt} |M_j|} \quad , \quad \underline{\nabla^- \rho_A} = \nabla \rho_{|_{M_{i-1}}} \quad , \quad \underline{\nabla^+ \rho_A} = \nabla \rho_{|_{M_i}}.$$

- In case of star volume control (see subsection 4.3.1), the **gradient computing** parameter permits to choose how the density gradient $\overline{\nabla \rho_A}$ is computed.

  > Either we take into account the density in the actual and in the upstream cell (then we need to identify this upstream cell, see Figure 5).

  > Or it is computed by taking by considering the value in the actual cell and the mean value in the surounding ones.

- A **flux limiter** ensures stable simulation and avoids spurious oscillations in the vicinity of the discontinuities (see [1]).
  By default, MinMod limiter is implemented but some others are disponible, see section 5.2 for more details.

- The **epsilon** criterion defines when the gradient is too hight between 2 cells, so, when the flux limiter is used.

4.4 Finite Elements parameters

In the GUI, two **FE schemes** are implemented in order to solve system (1b)-(1c) :

4.4.1 Constrained optimisation

It can be first considered as a constrained optimization problem, then a Gear scheme is implemented and the time discretization is :

$$\rho\left(\frac{3\mathbf{u}^{n+1}-4\mathbf{u}^{n}+\mathbf{u}^{n-1}}{2\Delta t}+\left((2*\mathbf{u}^{n}+\mathbf{u}^{n-1})\cdot\nabla_{\mathbf{x}}\right)\mathbf{u}^{n+1}\right)-\mu\Delta_{\mathbf{x}}\mathbf{u}^{n+1}+\nabla_{\mathbf{x}}p^{n+1}=\mathbf{f}^{n+1},$$
$$\mathrm{div}_{\mathbf{x}}\mathbf{u}^{n+1}=0.$$

The matrix expression of this sytem is so :

$$\begin{bmatrix}A & B^t\\ B & 0\end{bmatrix}\cdot\begin{bmatrix}\mathbf{u}\\ p\end{bmatrix}=\begin{bmatrix}\mathbf{f}(\mathbf{u})\\ \mathbf{f}(p)\end{bmatrix}$$

Where B represents the divergence term matrix and A the Non linear and the mass matrix.
The pressure term is considered as a Lagrange multiplier and the linear system is solved thanks to the Matlab direct inversion (backslash operator).

4.4.2 Projection method

It can also be solved by using a penalization-type projection method with, rotational incremental scheme, as presented in [5].
Main steps are the followings :

$$\frac{3\rho^{n+1}-4\rho^{n}+\rho^{n-1}}{2\Delta t}+(2*\mathbf{u}^{n}+\mathbf{u}^{n-1})\nabla\cdot(\rho^{n+1})=0,$$

$$\rho^{n+1}\frac{3\mathbf{u}^{n+1}-4\mathbf{u}^{n}+\mathbf{u}^{n-1}}{2\Delta t}+\rho^{n+1}(\mathbf{u}^{\star}\cdot\nabla)\mathbf{u}^{n+1}-\mu\Delta\mathbf{u}^{n+1}+\nabla\left(p^{n}+\frac{4}{3}\phi^{n}-\frac{1}{3}\phi^{n-1}\right)=\mathbf{f}(t^{n+1}),$$

$$\Delta\phi^{n+1}=\frac{3\chi}{2\Delta t}\nabla\cdot\mathbf{u}^{n+1},$$

$$p^{n+1}=p^{n}+\phi^{n+1}-\mu\nabla\cdot\mathbf{u}^{n+1}.$$

Remark 3 *Some other FE schemes are implemented, and also other solvers, as other projection methods and iterative methods (Conjugated Gradient and GMRES algorithms), see section 5.3 for more details.*

4.5 Computation parameters

- **Stop criterion** λ is used for defining if solution converges to stationary solution. This is described as $\frac{\|\mathbf{u}^{n}-\mathbf{u}^{n-1}\|}{\|\mathbf{u}\|}<\lambda$ and $\frac{\|p^{n}-p^{n-1}\|}{\|p^{n}\|}<\lambda$ and $\frac{\|\rho^{n+1}-\rho^{n}\|}{\|\rho^{n}\|}<\lambda$.

  This parameter is also used for defining if solution exploses in finite time. This is described as $\|u\|>1/\lambda$ or $\|p\|>1/\lambda$ or $\|\rho\|>1/\lambda$).

  Remark 4 *Here we consider $L^2(\mathbb{R}^2)$ norm.*

- User can choose between the four following **time splittings** :

  > Euler splitting
  > $(n+1)$ velocity linear second order extrapolation
  > $(n+1/2)$ velocity linear second order extrapolation
  > "Strang splitting" [7]

  NB : Time discretisation of those splittings are presented in appendix A.

- **Computation step time** dt will be computed as $dt = C \cdot h_{max}^{\alpha}$ with preset parameters :
  - $C = 1$ which refers to the mesh regularity.
  - $\alpha = 3/2$ which permits to obtain the good ratio between the order 3 time accuracy and the order 2 space accuracy.

4.6 Outputs parameters

- **Display** : for displaying figures during the computation. Simulation will be slower because of the graphical treatment.
- **Print results in a file** : errors and norms which are displayed in the Matlab command window can be printed in a text file, then, enter a directory name. Results will be printed in : "/OUTPUTS/directory_name/iterations_informations.txt".
- **Keep data for animation** : data can be saved for ploting an animation after the computation. In this case, enter the saving step time and a directory name. Data files will be created in : "/OUTPUTS/directory_name/
  ANIMATION/"

  See section 6 for more details.
- **Save assembling matrix** : assembling matrix can be saved for working on linear algebra after computation. In this case, enter the directory name and the saving step time. Matrix will be saved in :"/OUTPUTS/directory_name/MATRIX/".

Backups files are also saved for each simulation (see section 5.4 for more details). The "manual_setup.m" file. is created in the main workspace and a copy of this file is also created in the the "directory_name/" eventualy created (see in section 5 more details).
NB : Those parameters are grouped in the "OUTPUT" structure.

5 Advanced settings

The GUI permits to set several basics parameters, but few other ones can also be modified. When user starts a simulation from the GUI, some default parameters are declared in the same time (see part AA of the "AUXILIARY_FUNCTIONS/GUI_computation_settings.m" file). Those parameters permit to acces to other functionalities, schemes and methods.
Each time that a simulation is started from the GUI, a summarization of all parameters is done, including the hidden ones set by default. This file, called "manual_setup.m", is useful for running computation faster and also for accessing to all parameters.

NB : This file is writen thanks to the function "manual_setup_writting" located in "AUXILIARY_FUNCTIONS/".

5.1 Pressure constraint

As seen in section 2.2, an external constraint needs to be set to define pressure solution. By default, we set zero pressure to the nearest point of the center of the geometry. But, in case of a high pressure gradient in this region, the zero pressure point is placed in an other part of the geometry (a boundary for example).
Instead of imposing a zero pressure point, we can also set a zero pressure average condition in order to find pressure solution. This parameter is available in part "B-6" of manual_setup.m.

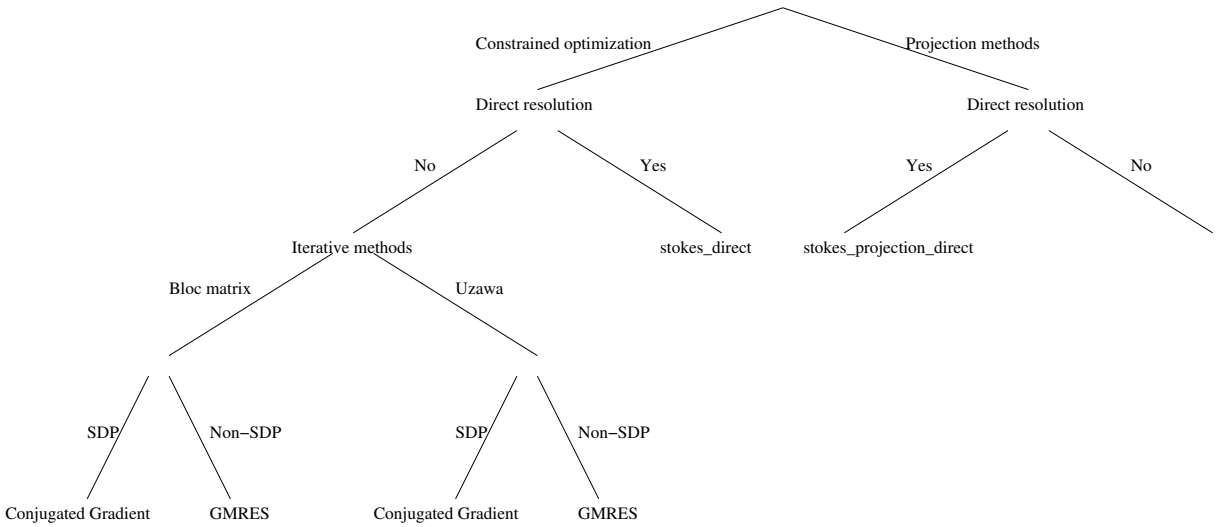


Figure 6: FE schemes tree.

5.2 Advanced FV parameters

5.2.1 Flux limiter

As presented in subsection 4.3.2, by default, a MinMod flux limiter ensures stable simulation and avoids spurious oscillations in the vicinity of the discontinuities (see [1] for more details).
The flowing flux limiters are also implemented :

- "Van Leer"
- "Van Albada"
- "Superbee"

Those flux limiters can be activated in function "function_limiter" of "COMPUTATION/FV/".

5.2.2 FV step time

As seen in section 4.5, the main step time is computed as $dt = C \cdot h_{max}^{\alpha}$ with preset parameters $C = 1$ and $\alpha = 3/2$. But the FV scheme is stable under the condition $dt \leq \dfrac{h_{max}}{\|\mathbf{u}\|_{\infty}}$.
So, in order to use a suitable step time, the FV step time is tooken as :

$$dt_{FV} = \min\left(dt_{FE},\ \frac{C_{FV} \cdot h_{max}}{\|\mathbf{u}\|_{\infty}}\right)$$

Where $C_{FV} = 0,1$ by default and h_{max} is the space parameter define in remark 1.
More details about the FV scheme implemented in NS2DVD are given in [3] and [2].

5.3 Advanced FE parameters

The Finite Element part can be solved by different ways than the ones seen in section 4.4. Figure 6 shows settings posibilities : The two methods which are available with the GUI are the ones in red on the scheme. Other methods are accessible by modifying the parameter "STOKES_DIRECT_INVERSION" from "yes" to "no" in the "FE PART" of the "manual_setup.m" file. Then, set the iterative methods parameters in part "B-6" of the same file.
Distinction between different methods is done in : "COMPUTATION/FE/resol_1it_ns_P1P2P1.m". Each ending branch represents a different function, some informations about algorithms used are given inside those files.

5.4 Rerun computation

In order to rerun a computation, some backup files are saved in : "OUTPUTS/BUFFER_DIRECTORY_FOR_BAC
or in : "/OUTPUTS/DIRECTORY_NAME/RERUN_BACKUPS/" depending if a directory has been created for the simulation or not (ie, if we save command window informations, if we save matrix or if we save solution for an animation, a new directory is automatically created).
Those files contain variables needed for reruning a simulation from a given iteration. For example, solutions time t^n, t^{n-1} and t^{n-2} for numerical schemes.
For reruning a computation, in "RERUN computation settings" part of the "manual_setup" file, switch "RERUN" variable from "no" to "yes". Then, run the file, a message box will open, select the backup file from which one you want to restart computation. The computation will stop again at the same "FINAL_TIME" than the first computation, so, for a longer computation, modify this parameter in "PHYSICAL SETTINGS" part.
The number of iterations between two backups can also be modify in part B-8 of the "manual_setup.m" (or in : "AUXILIARY_FUNCTIONS/GUI_computation_settings.m".

6 Post-processing

Some post-processing functions are implemented, depending of the benchmark case. Change directory to "POST_PROCESSING/" and run the corresponding function. Then, pick the directory name of the simulation you want to post-process.

6.1 Convergence rates

When analytical solution is known, the computed one can be compared to it. Then, if few simulations have been run with different edge sizes, convergence rate of error depending of the mesh size can be checked.

- For the analytical benchmark case (7.1.1), errors between speed, density and pressure analytical solution and the computed one is plotted.
- For the gaussian translation (7.1.3) or rotation (7.1.4), only density error depending of the mesh size is plotted because FE part is not computed.

For checking convergence rates of one of those 3 benchmark cases, in the GUI, check

6.2 Animation

If several iteration solutions have been saved (see OUTPUTS), then user can plot density islolines solutions in a figure by running ' 'plot_benchmark.m". User can also record a video of this simulation by running "video_benchmark.m".

7 Benchmark cases

The GUI is preset for reproducing some classical simulations, some of them permits to validates schemes. New benchmark cases can be implemented by following the procedure presented in subsection 7.3.

7.1 Validation benchmarks

7.1.1 Analytical Benchmark (EXAC)

This benchmark, presented in [3], permits to check convergence rates by evaluating the ability of a scheme to recover analytical solution.

The analytical solution is :

$$\left\{ \begin{array}{rcl} \rho_{\text{ex}}(t,x,y) & = & \rho_1(r,\theta-\sin t) \\ \mathbf{u}_{\text{ex}}(t,x,y) & = & \begin{pmatrix} y\cos t \\ x\cos t \end{pmatrix} \\ p_{\text{ex}}(t,x,y) & = & \sin x \sin y \sin t \end{array} \right.$$

Where $\rho_1(r,\theta) = 2 + r\cos\theta$ and where (r,θ) are the usual polar coordinates. The fields $\rho_{\text{ex}}(t,x,y)$ and $\mathbf{u}_{\text{ex}}(t,x,y)$ satisfy the mass conservation equation (1a) identically and $\mathbf{u}_{\text{ex}}(t,x,y)$ is solenoidal. The momentum equation (1b) is satisfied with the body force defined by :

$$\mathbf{f}_{\text{ex}}(t,x,y) = \begin{pmatrix} (y\sin t - x\cos^2 t)\rho_{\text{ex}}(t,x,y) + \cos x \sin y \sin t \\ -(x\sin t + y\cos^2 t)\rho_{\text{ex}}(t,x,y) + \sin x \cos y \sin t \end{pmatrix}$$

If computation is performed on a disk domain, preset boundary conditions are Dirichlet ones for velocity and pressure. Else, if domain is a square, boudary values are computed thanks to the exact solution.

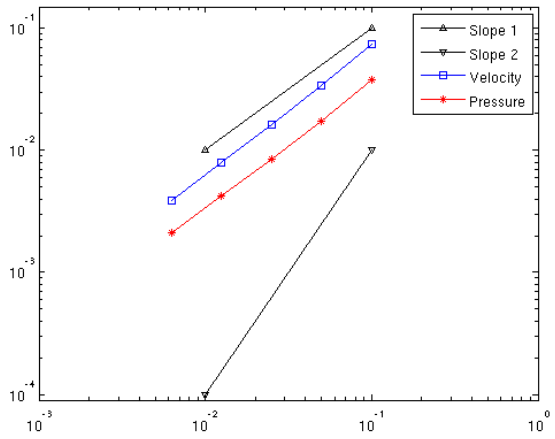


Figure 7: Convergence rate

7.1.2 Lid Driven Cavity (DCAV)

This benchmark permits to validate the finite elements scheme by observing that the intial constant density is preserved during the computation and that the results for the velocity and the pressure perfectly coincides with computations that use a standard Navier-Stokes code. Therefore, we can know if this coupling introduces or not spurious variation of density and degradation of the accuracy. On the top boundary, horizontal velocity is preset to $1m.s^{-1}$. For denstity and other velocity terms, boundary conditions are homogeneous Dirichlet ones.

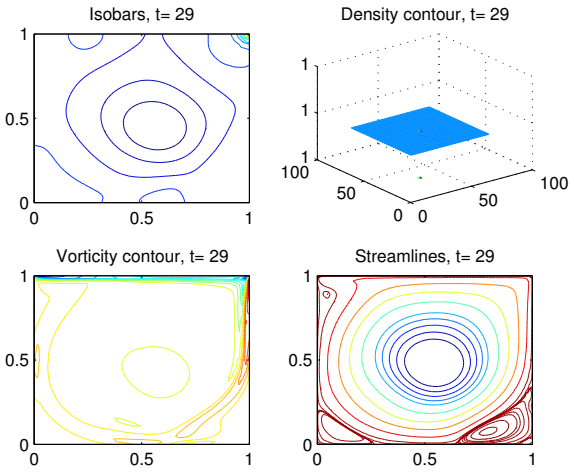


Figure 8: The lid-driven cavity test with a homogeneous density at initial time: Pressure, Density, Vorticity and Streamlines.

7.1.3 Density Gaussian Translation (TRHO)

This benchmark pemits to validate the finite volumes scheme. It represents the translation of a density gaussian exposed to a constant horizontal speed u. Boundary conditions are periodic on vertical sides and homogeneous Dirichlet on horizontal sides for density. As velocity field is imposed and not computed, the Finite Element part is not solved.
The domain geometry is preset to the square $[-0.25, 0.25]^2$ and the horizontal speed to 1 $m.s^{-1}$
Density exact solution is :

$$\rho_{ex} = 1 + \varphi_0 \cdot exp\Big(\frac{-\big(x - (x_0 + t)\big)^2 - (y - y_0)^2}{a^2}\Big)$$

with the folowing preset parameters :

- $x_0 = -0.5,\ y_0 = 0$ the gaussian initial coordinnates,
- $a = 0.15 \cdot \sqrt{2}$ the gaussian radius,
- φ_0 the gaussian amplitude.

NB : See in file "compute_sol_exacte.m" for changing parameters.

Check error between two same positions of the density gaussian on the domain for comparing, not the maximum error.

7.1.4 Density Gaussian Rotation (RRHO)

This benchmark is quite equivalent to the preceding one 7.1.3. Here, the density gaussian rotates counter-clockwise due to the speed field :

$$\mathbf{u} = \begin{pmatrix} -2y\cos(t) \\ +2x\cos(t) \end{pmatrix}$$

The preset domain geometry is $x_1 = -0.5,\ x_2 = 1.0;\ y_1 = -1.0,\ y_2 = 0.5$ and boundary conditions are Dirichlet homogeneous ones on every side for density. As in the Density Gaussian Translation case, due to the imposed velocity field, FE part is not used. Check error between two same positions of the density gaussian on the domain, not the maximum error.

Density exact solution is :

$$\rho_{ex} = 1+\varphi_0 \cdot exp\left[\frac{-\Big(x\cos\left(2\sin(t)\right)+y\sin\left(2sin(t)\right)-x_0\Big)^2-\Big(y\cos\left(2\sin(t)\right)-x\sin\left(2sin(t)\right)-y_0\Big)^2}{a^2}\right]$$

with the folowing preset parameters :

- $x_0 = 0,\ y_0 = -0.5$ the gaussian initial coordinates,
- $a = 3 \cdot 0.15 \cdot \dfrac{\sqrt{2}}{4}$ the gaussian radius,
- $\varphi_0 = 1$ the gaussian amplitude.

See in file "compute_sol_exacte.m" for changing parameters.

7.2 Other benchmarks

7.2.1 Rayleigh-Taylor Instability (RTIN)

This benchmark presented in [4] and [3] represents the evolution of two different density fluids. The heavier, located on the top, leaks in the light one, located on the bottom.

The interface is slightly smoothed since we set at time $t = 0$:

$$\rho_0(x,y) = \frac{\rho_{\mathrm{m}}+\rho_{\mathrm{M}}}{2} + \frac{\rho_{\mathrm{M}}-\rho_{\mathrm{m}}}{2}\tanh\Big(\frac{y-\eta\cos(2\pi x/d)}{0.01d}\Big),$$

with $\rho_{\mathrm{M}} > \rho_{\mathrm{m}} > 0$, and $\eta > 0$ the amplitude of the initial perturbation.

The preset domain geometry is $\Omega = (-d/2, d/2) \times (-2d, 2d)$, where $d = 1$. But, for reducing computation time, simulation can be done on $\Omega = (0, d/2) \times (-2d, 2d)$. Boundary condition on the top and bottom are Dirichlet ones and Neumann ones on vertical sides.

For reproducing this benchmark case, set the following parameters :

- $nbseg_x = 4k_1$ et $nbseg_y = 32k_2, \qquad k_1, k_2 \in \mathbb{N}$
- $Re = 1000$
- $T = 2$
- $\rho_{min} = 1$
- $\rho_{max} = 3$

On the following figure, one can see the evolution of the interface plotted thanks to the "plot_rtin_and_drop" function.

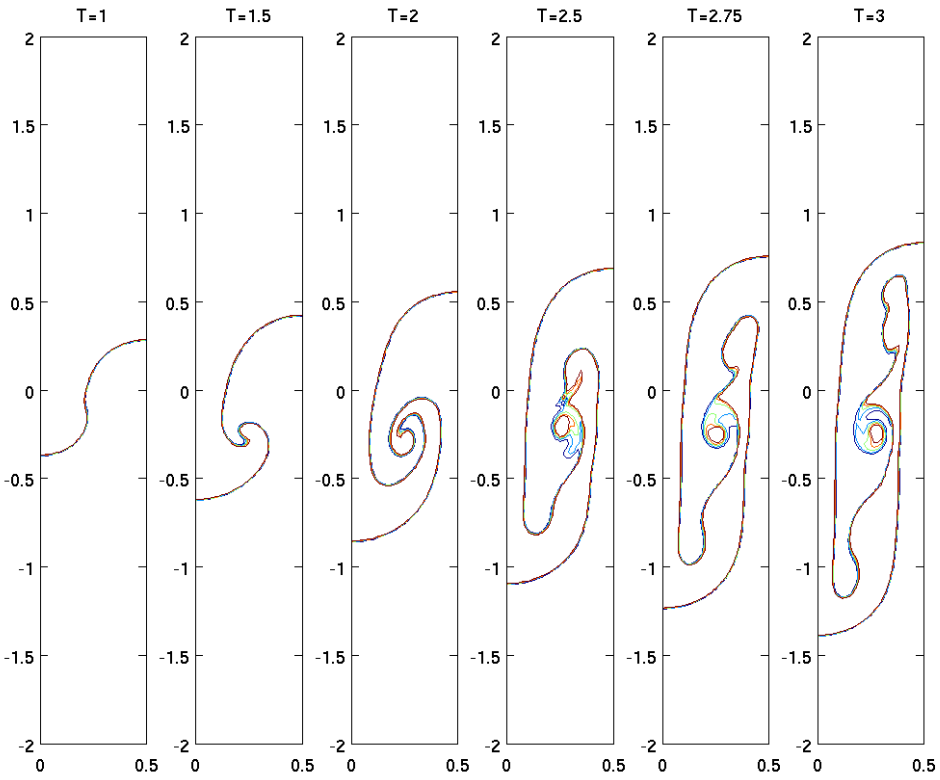


Figure 9: Rayleigh-Taylor instability: evolution of the interface; $Re = 5000$, density ratio $= 3$, initial amplitude $\eta = 0.1$. structured cross mesh 40×320, density contours $1.4 \leq \rho \leq 1.6$.

7.2.2 Falling Droplet (DROP)

This benchmark is inspired from [6]. A heavy "droplet" falls through a light fluid and impacts into the plane surface of the heavy fluid in a cavity. The computational domain is $(0, d) \times (0, 2d)$, where $d = 1$ and at $t = 0$ the fluid is at rest with density:

$$\rho(x, y) = \begin{cases} 100 & \text{if } 0 \leq y \leq 1 \text{ or } 0 \leq r \leq 0.2, \\ 1 & \text{if } 1 < y \leq 2 \text{ or } 0.2 < r, \end{cases}$$

where $r = \sqrt{(x - 0.5)^2 + (y - 1.75)^2}$. The volumic force in the momentum equation is $\rho * g$. Boundary condition are Dirichlet ones on every boundary for speed and density.

On the following figure, one can see the evolution of the interface plotted thanks to the "plot_rtin_and_drop" function.

- $nbseg_x = 6k_1$ et $nbseg_y = 10k_2$, $\quad k_1, k_2 \in \mathbb{N}$
- $T = 0.875$

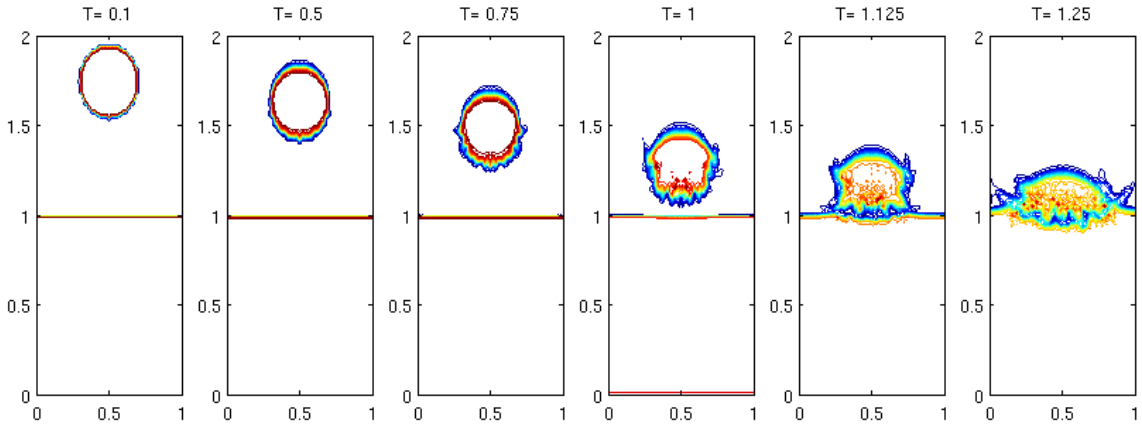


Figure 10: Falling droplet: evolution of the interface; $Re = 3132$, density ratio $= 100$, structured cross mesh 80×160, density contours $1.4 \leq \rho \leq 1.6$.

7.3 New benchmark cases

New benchmark cases can be implemented by following this brief procedure :

- Introduce specific parameters in benchmark_initialization.m if needed.
- Identify the boundary points in set_cl_P2P1.m
- If needed, identify the boundary points' number with Dirichlet conditions on density, see in set_cl_rho.m
- Define the pressure constraint
- Assemble "constant" matrix
- Set initial values in set_init_P1P2P1.m
- If it is known, implement the exact solution in function_sol_exact.m and compute_sol_exact_P1P2P1.m
- If needed, set Dirichlet boundary conditions on velocity in u_diri_P2.m
- Adapt the post-processing

Part III
Appendix

A Splitting time types

Let us denote Δt the time step and $t^n = n\,\Delta t$, $n \geq 0$ and let us assume that the numerical solution at time t^n, namely $(\rho^n, \mathbf{u}^n, p^n)$, is known on the computational domain. Four different splitting time type are implement in the NS2DDV code :

A.1 Euler splitting

This splitting time gives a time first order precision :

1. The new density field, ρ^{n+1}, is computed by solving on the time interval $(n\Delta t, (n+1)\Delta t)$ the transport equation
$$\partial_t \rho^{n+1} + \mathrm{div}_{\mathbf{x}}(\rho^{n+1}\mathbf{u}^n) = 0,$$
with suitable boundary conditions on ρ^{n+1}.

2. The new velocity and pressure fields, $\mathbf{u}^{n+1}$ and p^{n+1}, are computed by the resolution on the time interval $(n\Delta t, (n+1)\Delta t)$ of the system

$$\rho^{n+1}\big(\partial_t \mathbf{u}^{n+1} + \mathbf{u}^{n+1}\cdot\nabla_{\mathbf{x}}\mathbf{u}^{n+1}\big) + \nabla_{\mathbf{x}} p^{n+1} - \mu\Delta_{\mathbf{x}}\mathbf{u}^{n+1} = \mathbf{f}^{n+1},$$
$$\mathrm{div}_{\mathbf{x}}\mathbf{u}^{n+1} = 0,$$

completed by the specification of boundary conditions on $\mathbf{u}^{n+1}$.

Then we go back to the first step (using $n+1$ instead of n) to compute the solution at the following time steps.

A.2 $(n+1)$ velovity linear second order extrapolation

1. The new density field, ρ^{n+1}, is computed by solving on the time interval $(n\Delta t, (n+1)\Delta t)$ the transport equation

$$\partial_t \rho^{n+1} + \mathrm{div}_{\mathbf{x}}(\rho^{n+1}\mathbf{u}^*) = 0.$$

With $\mathbf{u}^* = 2\mathbf{u}^n - u^{n-1}$, which corresponds to the linear extrapolation of the velocity at time t^{n+1}.

2. The new velocity and pressure fields, $\mathbf{u}^{n+1}$ and p^{n+1}, are computed by the resolution on the time interval $(n\Delta t, (n+1)\Delta t)$ of the system

$$\rho^{n+1}\big(\partial_t \mathbf{u}^{n+1} + \mathbf{u}^{n+1}\cdot\nabla_{\mathbf{x}}\mathbf{u}^{n+1}\big) + \nabla_{\mathbf{x}} p^{n+1} - \mu\Delta_{\mathbf{x}}\mathbf{u}^{n+1} = \mathbf{f}^{n+1},$$
$$\mathrm{div}_{\mathbf{x}}\mathbf{u}^{n+1} = 0,$$

Then we go back to the first step (using $n+1$ instead of n) to compute the solution at the following time steps.

A.3 $(n+1/2)$ velovity linear second order extrapolation

1. The new density field, ρ^{n+1}, is computed by solving on the time interval $(n\Delta t, (n+1)\Delta t)$ the transport equation

$$\partial_t \rho^{n+1} + \mathrm{div}_{\mathbf{x}}(\rho^{n+1}\mathbf{u}^*) = 0.$$

With $\mathbf{u}^* = \frac{3\mathbf{u}^n - u^{n-1}}{2}$, which corresponds to the linear extrapolation of the velocity at time $t^{n+1/2}$.

2. The new velocity and pressure fields, $\mathbf{u}^{n+1}$ and p^{n+1}, are computed by the resolution on the time interval $(n\Delta t, (n+1)\Delta t)$ of the system

$$\rho^{n+1}\big(\partial_t \mathbf{u}^{n+1} + \mathbf{u}^{n+1}\cdot\nabla_{\mathbf{x}}\mathbf{u}^{n+1}\big) + \nabla_{\mathbf{x}} p^{n+1} - \mu\Delta_{\mathbf{x}}\mathbf{u}^{n+1} = \mathbf{f}^{n+1},$$
$$\mathrm{div}_{\mathbf{x}}\mathbf{u}^{n+1} = 0,$$

Then we go back to the first step (using $n+1$ instead of n) to compute the solution at the following time steps.

A.4 Strang splitting

1. The new density field, ρ^{n+1}, is computed by solving on the time interval $(n\Delta t, (n+1)\Delta t)$ the transport equation

$$\partial_t \rho^{n+1} + \mathrm{div}_{\mathbf{x}}(\rho^{n+1}\mathbf{u}^n) = 0,$$

with suitable boundary conditions on ρ^{n+1}.

2. The new velocity and pressure fields, $\mathbf{u}^{n+1}$ and p^{n+1}, are computed by the resolution on the time interval $(n\Delta t, (n+1)\Delta t)$ of the system

$$\rho^{n+1}\big(\partial_t \mathbf{u}^{n+1} + \mathbf{u}^{n+1}\cdot\nabla_{\mathbf{x}}\mathbf{u}^{n+1}\big) + \nabla_{\mathbf{x}} p^{n+1} - \mu\Delta_{\mathbf{x}}\mathbf{u}^{n+1} = \mathbf{f}^{n+1},$$
$$\mathrm{div}_{\mathbf{x}}\mathbf{u}^{n+1} = 0,$$

completed by the specification of boundary conditions on $\mathbf{u}^{n+1}$.

3. The following velocity and pressure fields, $\mathbf{u}^{n+2}$ and p^{n+2}, are computed by the resolution on the time interval $((n+1)\Delta t, (n+2)\Delta t)$ of the system

$$\rho^{n+1}\big(\partial_t \mathbf{u}^{n+2} + \mathbf{u}^{n+2}\cdot\nabla_{\mathbf{x}}\mathbf{u}^{n+2}\big) + \nabla_{\mathbf{x}} p^{n+2} - \mu\Delta_{\mathbf{x}}\mathbf{u}^{n+2} = \mathbf{f}^{n+2},$$
$$\mathrm{div}_{\mathbf{x}}\mathbf{u}^{n+2} = 0,$$

completed by the specification of boundary conditions on $\mathbf{u}^{n+2}$.

4. Finally, the new density field, ρ^{n+2}, is computed by solving on the time interval $((n+1)\Delta t, (n+2)\Delta t)$ the transport equation

$$\partial_t \rho^{n+2} + \mathrm{div}_{\mathbf{x}}(\rho^{n+2}\mathbf{u}^{n+2}) = 0,$$

with suitable boundary conditions on ρ^{n+2}.

Then we go back to the first step (using $n+2$ instead of n) to compute the solution at the following time steps.

References

[1] C.H. Bruneau and P. Rasetarinera. A finite volume method with efficient limiters for solving conservation laws. *Internal report 96024, MAB Laboratory, Bordeaux 1 University, France*, 1996. 7, 10

[2] C. Calgaro, E. Chane, E. Creusé, and T. Goudon. L1 stability of vertex-based muscl finite volume schemes on unstructured grids ; simulation of incompressible flows with high density ratios. *J. Comput. Phys.*, 229(17). 2, 5, 10

[3] C. Calgaro, E. Creusé, and T. Goudon. An hybrid finite volume-finite element method for variable density incompressible flows. *J. Comput. Phys.*, 227:4671–4696, 2008. 2, 5, 10, 11, 14

[4] Y. Fraigneau, J.-L. Guermond, and L. Quartapelle. Approximation of variable density incompressible flows by means of finite elements and finite volumes. *Commun. Numer. Methods Eng.*, 17:893, 2001. 14

[5] J.-L. Guermond and A. Salgado. A splitting method for incompressible flows with variable density based on a pressure poisson equation. *J. Comput. Phys.*, 228:2834–2846, 2009. 8

[6] T. Schneider, N. Botta, K. J. Geratz, and R. Klein. Extension of finite volume compressible flow solvers to multidimensional, variable density zero Mach number flows. *J. Comput. Phys.*, 155:248–286, 1999. 15

[7] G. Strang. On the construction and comparison of difference schemes. *SIAM J. Numer. Anal.*, 5:506–517, 1968. 8